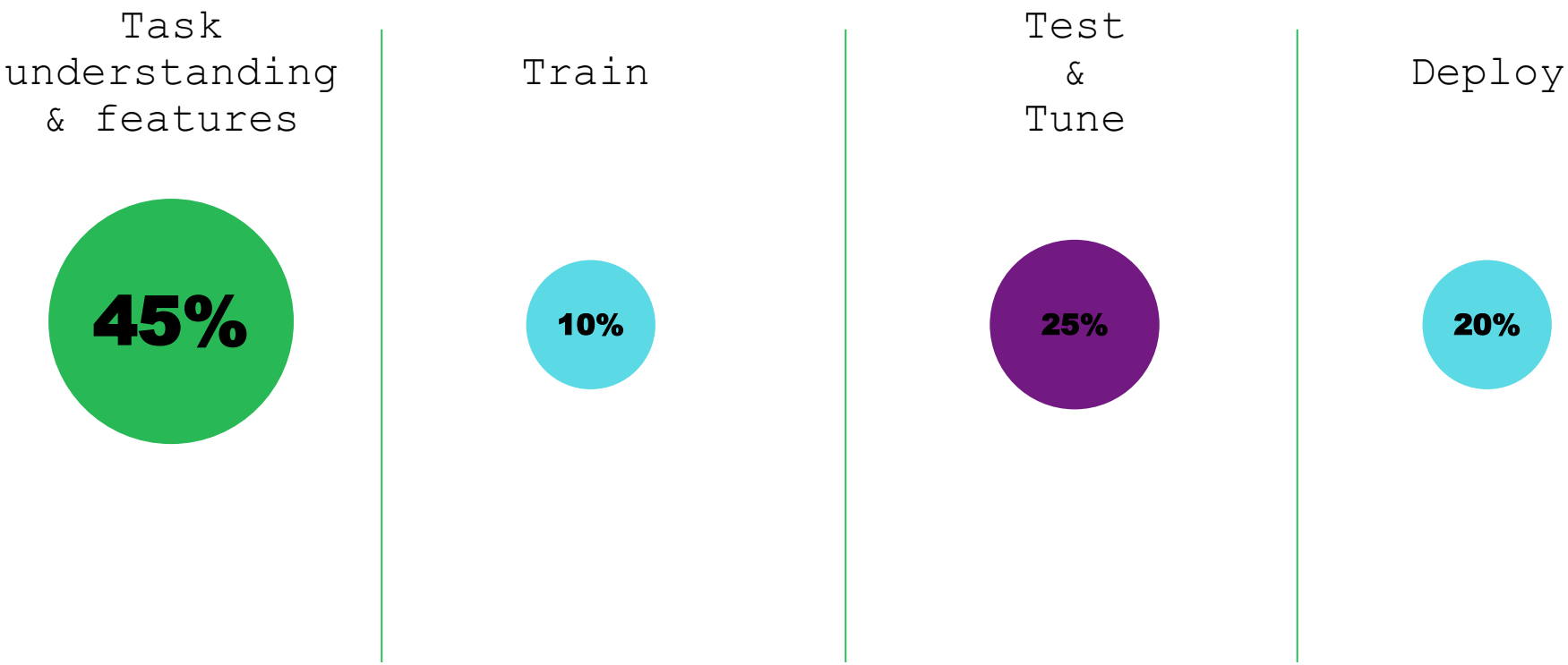
Optimize it!

Seleznev Artem, 24 June 2019

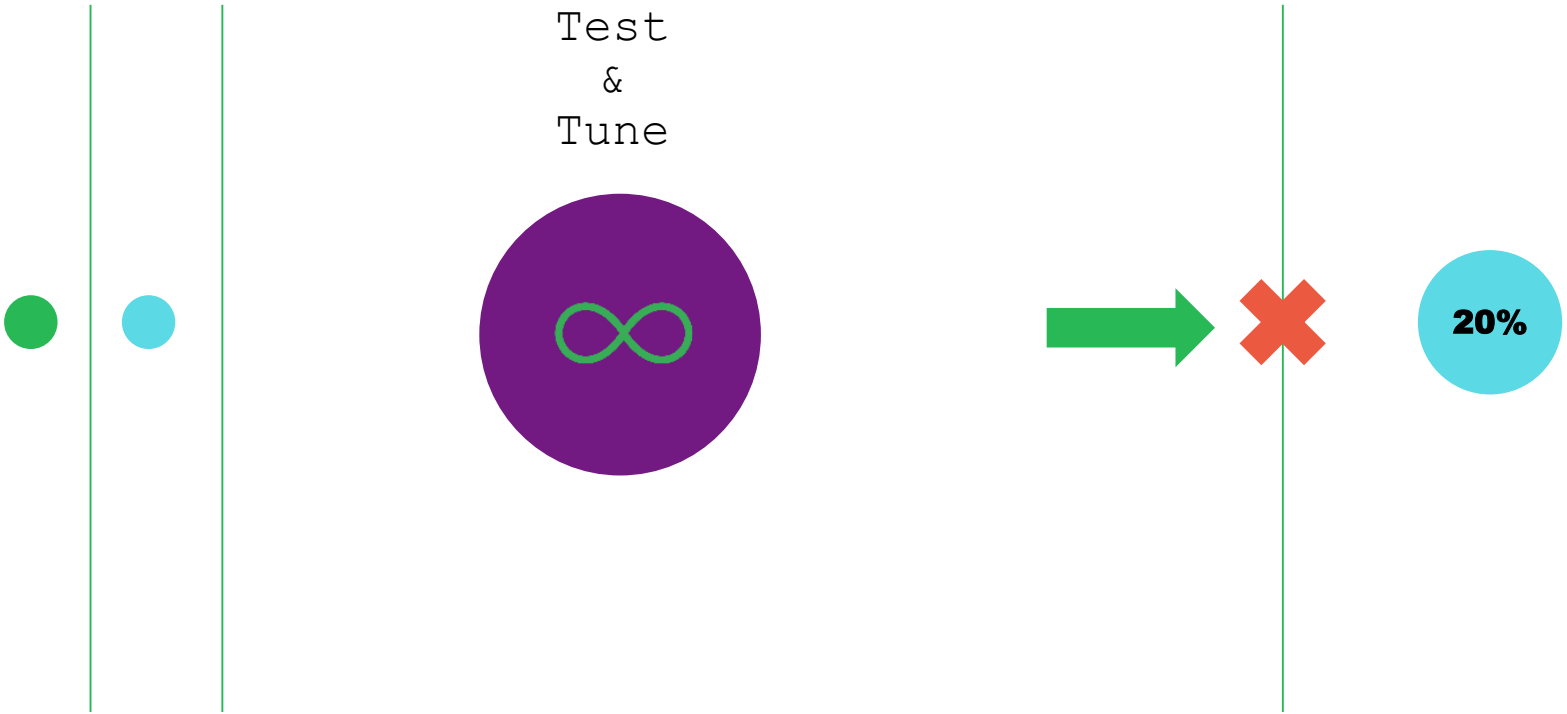
МегаФон

? < Time to market

? < Time to market



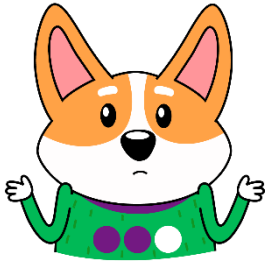
? < Time to market



Problems?

1. Local minimum (act. AUC = 0.71 => after OPT AUC = 0.69)

Problems?



1. Local minimum (act. AUC = 0.71 => after OPT AUC = 0.69)

2. Speed (more than 18.000 of significant parameters
and more 962.000.000 others)

```
{'n_estimators':  hp.choice('n_estimators', np.arange(200, 1000, 200, dtype = int)),
 'learning_rate': hp.uniform('learning_rate', 0.0001, 0.1 ),
 'max_depth': hp.choice('max_depth', np.arange(6, 10, 4, dtype = int)),
 'l2_leaf_reg' : hp.choice('l2_leaf_reg',np.arange(3, 7, 2, dtype = int)),
 'bagging_temperature':  hp.choice('bagging_temperature',np.arange(0, 10, 1, dtype = int)),
 'random_strength' : hp.uniform('random_strength', 0.001, 0.9 ) }
```

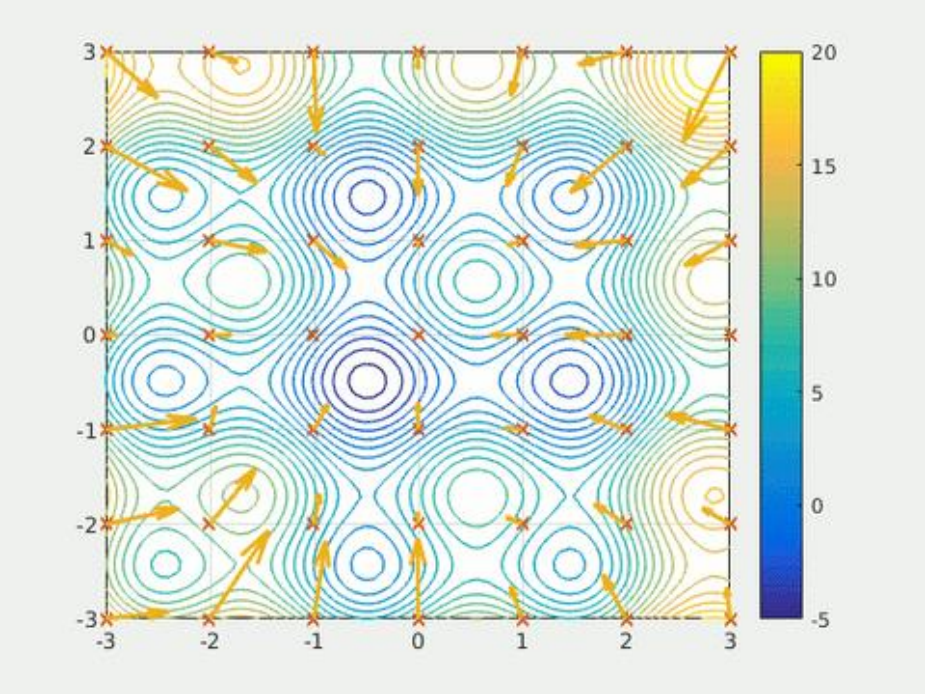

Best
Better
Good
How can it be improved?

SWARM! ! !

SWARM!!!

Complex algorithm:

- Count of Agents
- Current velocity
- Local velocity
- Global velocity
- Loss function



Add ~~problems~~ a little speed

1.
2.

Add ~~problems~~ a little speed

1. Fitting falls
2.

Exceptions:

1. Memory exception
2. Allocate memory

Add ~~problems~~ a little speed

1. Fitting falls
2. Exceptions

```
async def handle_exception():
    try:
        await bug()
    except Exception:
        print('You have a bug!')

async def bug():
    raise Exception()

loop = asyncio.get_event_loop()
loop.create_task(handle_exception())
loop.run_forever() loop.close()
```

Async/await

```
import yaopt
from yaopt import yaopt
from sklearn.metrics import roc_auc_score

CNT_SPLITS = 3

async def one_fit(alg, param, set_x, set_y):
    alg.set_params(**param)
    alg = await alg.fit(set_x, set_y)
    predict_auc = roc_auc_score(test_y, alg.predict_proba(test_x))
    if best_auc < predict_auc:
        await alg
```

Async/await

```
import yaopt
from yaopt import yaopt
from sklearn.metrics import roc_auc_score

CNT_SPLITS = 3

async def one_fit(alg, param, set_x, set_y):
    alg.set_params(**param)
    alg = await alg.fit(set_x, set_y)
    predict_auc = roc_auc_score(test_y, alg.predict_proba(test_x))
    if best_auc < predict_auc:
        await alg


async def multifitter(alg, param, best_auc, set_x, set_y, test_x, test_y):
    async with yaopt.open() as thrd:
        thrd.start(one_fit, alg, param, set_x, set_y)
    return thrd.found()
```

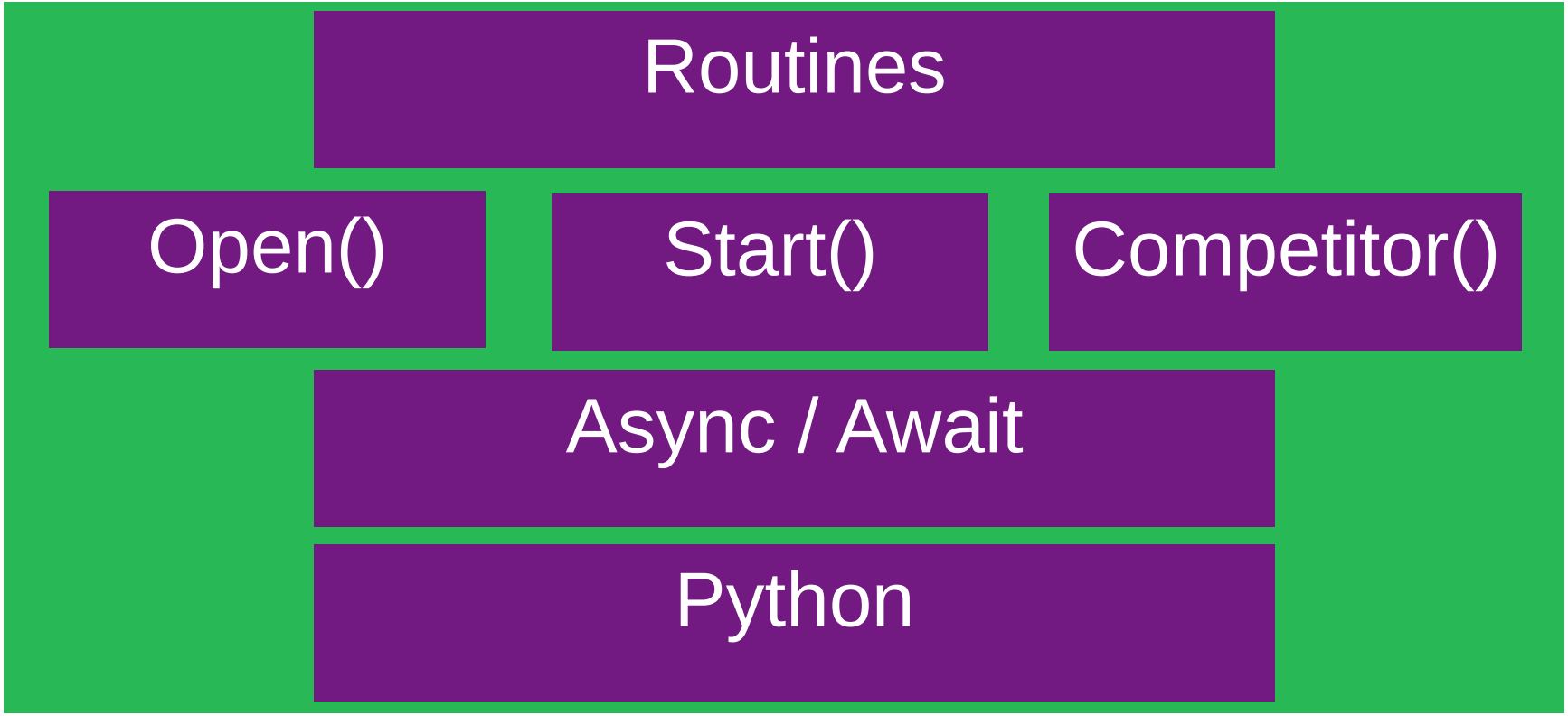
Async/await

```
async def main():
    status = await yaopt.optdriver(alg, params, set_x, set_y)
    if status:
        with yaopt.manager(CNT_SPLITS):
            alg, param, best_auc, set_x, set_y, test_x, test_y = yaopt.base(alg,
                                                                 params, set_x, set_y)


            best_alg = await multifitter(alg, param, best_auc,
                                         set_x, set_y, test_x, test_y)
    print('Best param was found')

yaopt.run(main)  # activator
```

Async/await

```
async def main():
    status = await yaopt.optdriver(alg, params, set_x, set_y)
    if status:
        with yaopt.manager(CNT_SPLITS):
            alg, param, best_auc, set_x, set_y, test_x, test_y = yaopt.base(alg,
                                                                params, set_x, set_y)


            best_alg = await multifitter(alg, param, best_auc,
                                         set_x, set_y, test_x, test_y)
    print('Best param was found')

yaopt.run(main)  # activator
```

Async/await




Under the hood

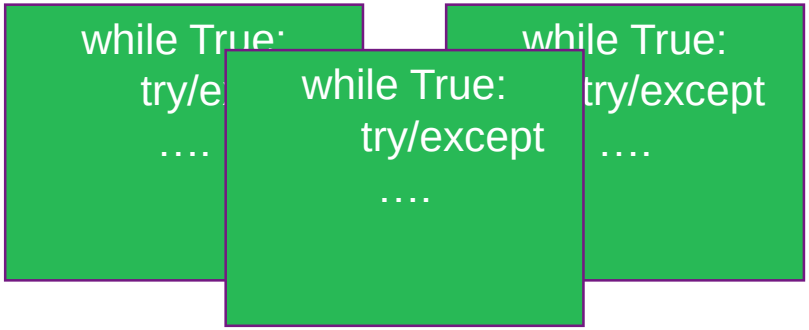
Basic primitives

asyncio:

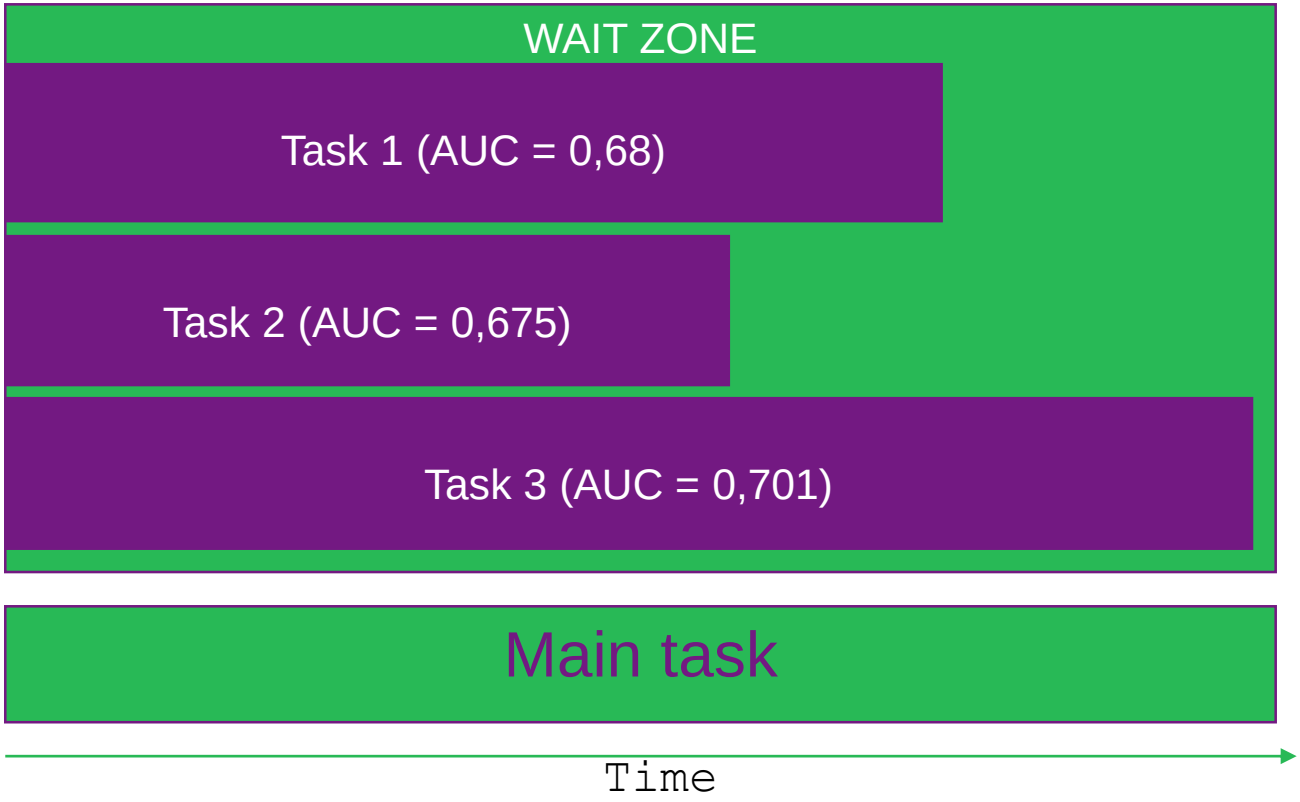
```
add_done_callback()
ensure_future()
transport.write()
loop.add_reader()
```

…

```
async def multifitter(…):
    async with yaopt.open()
```



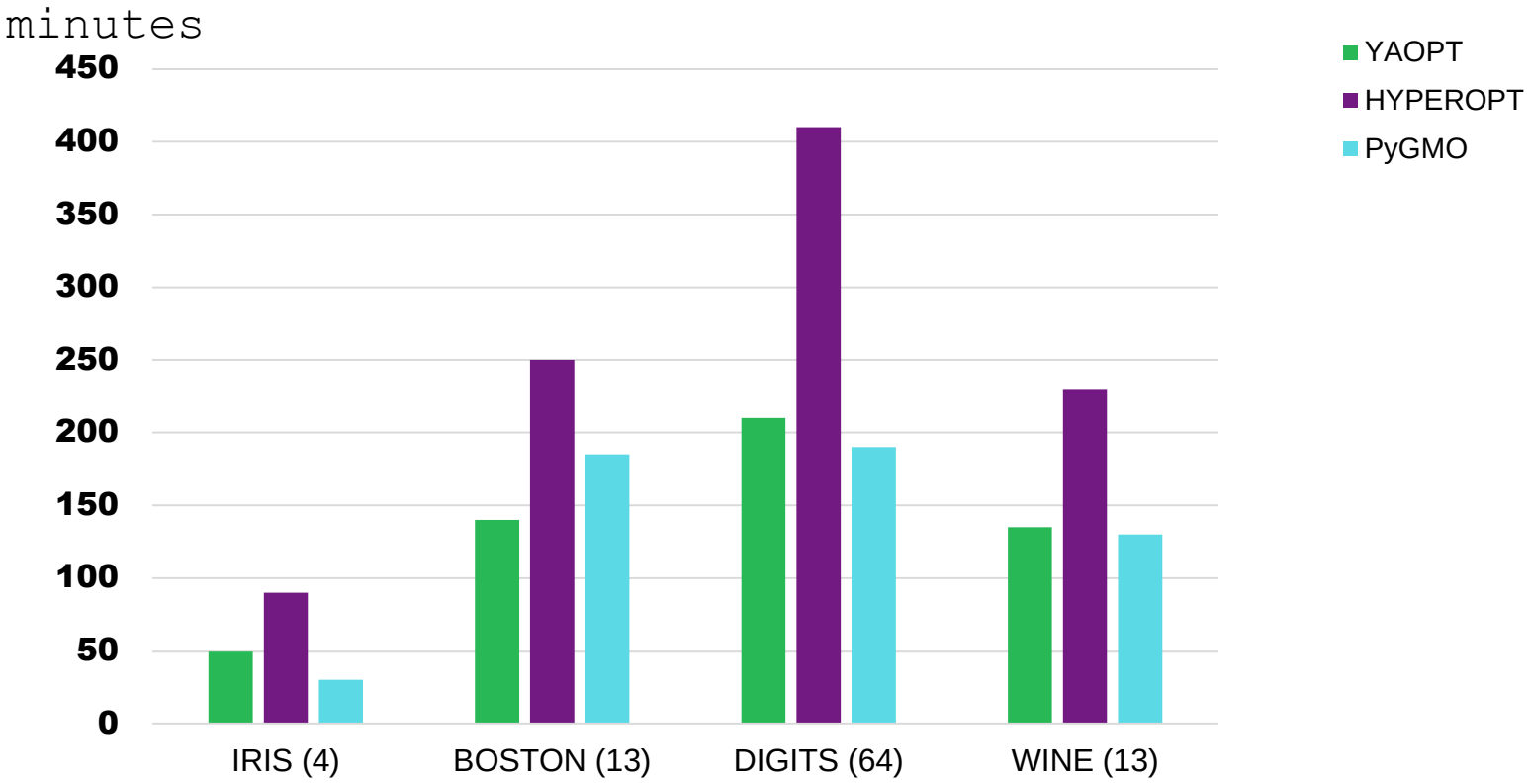
Start and Competitor




```
...

yaopt.competitor(d)


d = difference

d = %
d = N
```


Result

Result


minutes
450
400
350
300
250
200
150
100
50
0
IRIS (4)
BOSTON (13)
DIGITS (64)
WINE (13)
YAOPT
HYPEROPT
PyGMO

The best optimizer begins with you

fb: /seleznev.artem.info
telegram: @SeleznevArtem